REMARKS

Pursuant to the Request for Continued Examination (RCE) filed August 5, 2008, and in response to the Final Office Action mailed September 24, 2007, and the Notice of Non-Responsive amendment mailed October 16, 2008, Applicants respectfully request reconsideration. Claims 1-8 and 12 were previously pending in this application. By this amendment, claims 1, 5, 7 and 8 have been amended. As a result, claims 1-8 and 12 are pending for examination with claims 1, 5, 7, and 8 being independent. No new matter has been added.

Response to Interview Summary

As a preliminary matter, Applicants' representatives thank Examiner Kiss for the courtesies extended in granting and conducting a telephone interview held on April 17, 2008. During the telephone interview, possible claim amendments were discussed. As a result, Applicants have amended independent claims 1, 5, 7 and 8 in accordance with the discussion.

Rejections Under 35 U.S.C. §103

The Office Action rejected claims 1-8 and 12 under 35 U.S.C. 103(a) as allegedly being unpatentable over Hadjiyiannis et al., "ISDL: An Instruction Set Description Language for Retargetabillity" (hereinafter "Hadjiyiannis"), further in view of Vos, GB 2,127,188 (hereinafter "Vos"), and further in view of A. N. Edmonds "Microcoding and bit-slice techniques" (hereinafter "Edmonds"). Applicants respectfully disagree. In addition, Applicants have amended independent claims 1, 5, 7, and 8 in accordance with the discussion during the telephone interview. During the interview, the Examiner appeared to agree that cited references do not teach that the assembler automatically tracks changes in the instruction set by acquiring the data from the descriptor file reflecting the changes thereby a hard-coding of the assembler is reduced.

A. Independent Claim 1

Claim 1, as amended, recites:

> An assembler for a target microprocessor, the assembler comprising:
> a descriptor file containing information descriptive of the instruction set of said target microprocessor, wherein information about a starting position of at least

one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set;

a translation device of the assembler for translating assembly language instructions into machine language as an output;

a fetching device of the assembler for deriving information from the assembly language instructions and acquiring data from said descriptor file using the information, wherein the data is representative of the constraints due to the instruction set, and wherein *the assembler automatically tracks changes in the instruction set by acquiring the data from the descriptor file reflecting the changes thereby a hard-coding of the assembler is reduced*;

a control device of the assembler arranged to receive said data from said fetching device and said machine language from said translation device, and operable to constrain the machine language to conform to an architecture of said instruction set using the data received from the fetching device; and

a data transfer device of the assembler arranged to output the selected data fetched from said descriptor file directly to a linker to enable the linker to perform operations on the constrained machine language using the selected data.
(Emphasis added).

On page 6, the Office Action alleges that Hadjiyiannis discloses "an assembler for a target microprocessor to automatically track changes in an instruction set of the target microprocessor, the assembler comprising ..." "tracking the changes is interpreted as being representative of constrains due to the instruction set architecture provided by the Instruction Set Description Language" of Hadjiyiannis. Hadjiyiannis does not disclose **automatically** tracking changes in an instruction set of the target microprocessor (emphasis added). Applicants have amended claim 1 to recite " the assembler automatically tracks changes in the instruction set by acquiring the data from the descriptor file reflecting the changes thereby a hard-coding of the assembler is reduced," as discussed during the telephone interview. Support for this amendment can be found on page 8, lines 31-32, and one page 9, lines 34-36 of Applicants' specification.

Hadjiyiannis does not disclose *an assembler* for a target microprocessor to *automatically track changes* in an instruction set of the target microprocessor (emphasis added). Hadjiyiannis is directed to a language called ISDL (Instruction Set Description Language) that can be used to describe target architectures in a retargetable **compiler** (emphasis added). Hadjiyiannis describes that the ISDL description of a target architecture can be used to **generate** an assembler (Section III,

paragraph 1, lines 7-8) (emphasis added). Hadjiyiannis discusses generating a compiler that produces assembly code specific to, and optimized for the target processor (Section III, paragraph 1, lines 5-7).

Further, Hadjiyiannis states that an automatic assembler **generator** was designed and implemented (Section V, paragraph 2, lines 1-2) (emphasis added). The automatic assembler **generator** receives an ISDL description as input, and **produces** an assembler which assembles the compiler's output to a binary file (Hadjiyiannis, Section V, paragraph 2, lines 2-3) (emphasis added). Therefore, while the assembler of Hadjiyiannis is generated automatically, nowhere does Hadjiyiannis even mention that this assembler is capable of **automatically tracking changes** in an instruction set (emphasis added). Rather, the assembler of Hadjiyiannis, which is an output of the retargetable compiler, appears to be fixed. If any changes occur, the ISDL description of a target architecture can be used to generate **another assembler**, as opposed to providing the assembler that automatically tracks changes in the instruction set by acquiring the data from the descriptor file reflecting the changes thereby a hard-coding of the assembler is reduced, as recited in claim 1.

In fact, Hadjiyiannis describes only that the assembler generator produces Lex and Yacc files which, when compiled, result in an executable capable of parsing the assembly [code] and generating the instruction words (Section V, paragraph 3, lines 1-3). This is the only portion where Hadjiyiannis describes the generated assembler.

On page 6, the Office Action alleges that Hadjiyiannis discloses "assembler comprising: a descriptor file containing information descriptive of the instruction set of said target microprocessor." However, nowhere does Hadjiyiannis state that the as*sembler* comprises a descriptor file (emphasis added). Hadjiyiannis describes that the automatically generated assembler transforms the code produced by the compiler to a binary file that is used as input to the Instruction Level Simulator (ILS). Therefore, the assembler of Hadjiyiannis *created using the ISDL description* is used to produce a binary code from an assembly code *specific to the target processor*. Thus, the assembler which is itself specific to this target processor takes as input the assembly code that has already been made specific to the target processor and produces, from the assembly code, a binary code to decouple the development of the compiler from that of the ISL, as discussed in Section V of Hadjiyiannis. Hadjiyiannis notes that the availability of the assembler allows assembly programs to

be written and tested on the ILS, even when no compiler is available (Section V, paragraph 1, lines 4-5). Therefore, the assembler of Hadjiyiannis does not include "a descriptor file containing information descriptive of the instruction set of said target microprocessor, wherein information about a starting position of at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set," as described in claim 1. Indeed, the assembler of Hadjiyiannis was generated using the ISDL description.

On page 7, the Office Action states that Hadjiyiannis discloses "wherein information about... at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set." Thus, the Office Action concedes that Hadjiyiannis does not disclose "information about a starting position and at least one bit field ... ."

Further, on page 7, the Office Action states Hadjiyiannis discloses a fetching device of the assembler for deriving information from the assembly language instructions and acquiring data from said descriptor file using the information. Again, the assembler of Hadjiyiannis was *generated* using the ISDL description and, contrary to the assertions made in the Office Action, does not acquire "data from said descriptor file using the information" (emphasis added). Furthermore, Hadjiyiannis does not teach or suggest that "the assembler automatically tracks changes in the instruction set by acquiring the data from the descriptor file reflecting the changes thereby a hard-coding of the assembler is reduced," as recited in claim 1. Indeed, in Hadjiyiannis, the assembler is specifically generated for particular target hardware architecture and therefore does not track any changes in the instruction set. In fact, a different assembler of Hadjiyiannis would need to be generated for another target processor.

On page 8, the Office Action states that the compiler of Hadjiyiannis is "part of the claimed assembler." Applicants respectfully disagree. One of skill in the art would appreciate that the compiler cannot be part of the assembler. An assembler is used to translate assembly language code into a machine code, while a compiler is used to translate high-level language instructions (e.g., C or C++ source program discussed in Hadjiyiannis) into a machine code. Thus, it appears that the compiler of Hadjiyiannis cannot be a part of an assembler at least for the reason that assembly

language code includes lower level instructions. Moreover, Hadjiyiannis describes both a compiler and an assembler which again supports the notion that the two are different.

The Office Action alleges that the compiler of Hadjiyiannis is "part of the claimed assembler in the sense that the respective mapped functions (e.g., translate, acquiring data from the instruction set and result (machine language) are equivalent as claimed)." However, with respect to "translate," claim 1 recites *a translation device of the assembler for translating assembly language instructions into machine language as an output* (emphasis added). In Hadjiyiannis, the compiler produces assembly code as shown in Fig. 1. Thus, the compiler of Hadjiyiannis does not perform the "translate" function, as stated in the Office Action. Furthermore, Hadjiyiannis states that the compiler generates *assembly code* (emphasis added). It is the assembler of Hadjiyiannis that is specifically generated to produce machine code. Thus, the compiler of Hadjiyiannis does not produce the "result (machine language)" either.

On page 8 and 9, the Office Action alleges that "it would have been obvious to one of ordinary skill in the art to constrain the machine language to conform to the architecture of said instruction set, instead of restraining the assembly language to the architecture, and thereby restraining the machine language, as two methods produce the same result." It appears that, by this statement, the Office Action equates compilers and assemblers and states that everything done using an assembler is "obvious" because it can be done using a compiler since both produce machine code. Applicants respectfully disagree. One of skill in the art would appreciate a difference between a compiler and an assembler. Moreover, it appears that "restraining the assembly language to the architecture, and thereby restraining the machine language" is not "obvious" which is supported by Hadjiyiannis where the machine language is not merely "thereby restrained" because the assembly language is restrained to the architecture, but a specific assembler is generated using the ISDL description to assemble the output of the compiler into the binary code. In addition, Hadjiyiannis describes that "the output of the compiler can be in assembly which is much more human readable than binary. Thus, some debugging can be performed without the use of an ILS." (Section 5, paragraph 1). Therefore, it may be suggested that the output of Hadjiyiannis' compiler may be *machine* code as well, in which case Hadjiyiannis does not describe restraining *the assembly language* to the architecture, as alleged in the Office Action (emphasis added).

On page 9, the Office Action concedes that Hadjiyiannis does not disclose "a data transfer device of the assembler arranged to output the selected data fetched from said descriptor file directly to a linker to enable the linker to perform operations on the constrained machine language using the selected data," as recited in claim 1. The Office Action goes on to allege that Vos discloses "...a data transfer device of the assembler arranged to output the selected data fetched from said descriptor file directly to a linker..." (E.g., see Figure 1, block10 + block 12 & page 2, lines 21-34), wherein *the linker command file* (block 12) is generated for the particular prototype processor and *the configuration object file* (block 10) includes interrupt vectors and procedures, memory configuration which are interpreted as instruction set information. Furthermore, the configuration file and linker command file are directly input to the linker" (emphasis added). Vos is directed to a system for translating hardware/software interface specifications simultaneously into specific microprocessor executable code and commands for the linker/loading system of a selected hardware configuration (page 1, lines 5-9). Vos does not discuss fetching "from said descriptor file" wherein a descriptor file is a file "containing information descriptive of the instruction set of said target microprocessor," as recited in claim 1. Moreover, Vos discusses that the Integration Control System (ISC) integration source file (ICS) (block 6) provides a concise, *human-readable description* of the *hardware/software interface* (page 2, lines 38-40) (emphasis added). The human-readable description of the *hardware/software interface* is different from recited in claim 1 "information descriptive of the instruction set of said target microprocessor."

Further, the Office Action states that "Hadjiyiannis and Vos are analogous art because they are both concerned with the same field of endeavor, namely, an architecture that is modifiable by input and adapts source code to such input and correspondingly outputs machine language. Therefore, at the time the invention was made, it would have been obvious to a person of ordinary skill in the art to utilize Vos' data capture device in Hadjiyiannis's system of claim 1 as an alternate method to implement architectural specifications. The motivation for doing so would have been to have a simpler design for a particular system, where Vos' method may be more efficient than Hadjiyiannis's for a particular objection." Applicants respectfully note that it is not clear in what way "Vos' method may be more efficient than Hadjiyiannis's for a particular objection." Vos is directed to the ISC system that provides an automatic method for *formatting a machine independent*

program written in a high level language (emphasis added). At the same time, Hadjiyiannis is directed to the *machine description language* used to describe target architectures to a retargetable compiler (emphasis added). Vos describes a Pascal language. In Vos, in linker 18, the configuration object file is linked to the Pascal object files and the run-time support library. It is not clear how one of skill in the art would combine machine description language of Hadjiyiannis used to describe target architectures to a retargetable compiler with the linker command file and the configuration object file of Vos to produce "a simpler design," as stated in the Office Action.

The Office Action states that Hadjiyiannis and Vos "are analogous art because they are both concerned with the same field of endeavor, namely, an architecture that is modifiable by input." However, while Vos provides formatting a *program* written in a *high level language*, particularly the Pascal language, Hadjiyiannis describes an automatic assembler generator. The only high level language mentioned in Hadjiyiannis is a C or C++ code used as input to the compiler. Moreover, a person of skill in the art would not refer to Vos describing the software/hardware integration system for the Pascal language "as an alternate method to implement architectural specifications" to "device in Hadjiyiannis's system," as alleged in the Office Action. Thus, contrary to the assertions made in the Office Action, one of skill in the art would not be motivated to combine Hadjiyiannis and Vos. Therefore, the alleged combination is improper.

On page 10, the Office Action concedes that neither Hadjiyiannis nor Vos discloses "... a starting position ... (E.g., see Fig. 1 & "Definition Facilities"), wherein the position and size of a field within the bitmap are disclosed." The Office Action then alleges that Edmonds discloses this limitation. However, Edmonds describes the pipelining skews in a meta assembly definitions file (page 267). While Edmonds states that the meta assembler can assemble code for any destination hardware and must therefore be fed with specific details about the hardware (page 266), the microcoding of Edmonds is different from both Hadjiyiannis and Vos and limitations of claim 1. In the "Definition Facilities" section, Edmonds discusses that common features of the assemblers for the definition process are the position and size of a field within the bit map, its default value, and the values that it can hold (page 267). However, these features are features of the meta assemblers rather than a descriptor file containing information descriptive of the instruction set of the target microprocessor recited in claim 1. Contrary to the assertions made in the Office Action, one of skill

in the art would not use "a starting position" of Edmonds that describes microcoding and bit-slice techniques to combine Edmonds with an alleged combination of Hadjiyiannis and Vos, which is already a highly unlikely combination, to obtain "a descriptor file containing information descriptive of the instruction set of said target microprocessor, wherein information about a starting position of at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set," as recited in claim 1.

In view of the forgoing, claim 1 patentably distinguishes over Hadjiyiannis, Vos, and Edmonds, either alone or in combination.

Claims 2-4 and 12 depend from claim 1 and are allowable for at least the same reasons.

Accordingly, withdrawal of the rejection of claims 1-4 and 12 is respectfully requested.

B. Independent Claim 5

Claim 5, as amended, recites:

> A method of assembling a machine language program for a target microprocessor comprising:
>
> providing a descriptor file containing information descriptive of an instruction set of said target microprocessor, wherein information about a starting position of at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set, and wherein the descriptor file reflects changes in the instruction set as the changes occur;
>
> translating assembly language instructions into machine language, wherein the translation comprises:
>
> directly transliterating the assembly language instructions into machine language;
>
> deriving information from the assembly language instructions and acquiring data from said descriptor file using the information, wherein the data is representative of the constraints due to the instruction set;
>
> *automatically tracking the changes in the instruction set by acquiring the data from the descriptor file thereby a hard-coding for the assembling is reduced*; and
>
> constraining the directly transliterated machine language to conform to an architecture of said instruction set, thereby assembling the machine language program for the target microprocessor, using the data acquired from the descriptor file; and

transferring the selected data acquired from said descriptor file directly to a linker to enable the linker to perform operations on the constrained machine language using the selected data.
(Emphasis added).

As should be clear from the above discussion, Hadjiyiannis, Vos, and Edmonds do not teach or suggest all of the limitations of claim 5. Specifically, none of the cited references teaches or suggests "providing a descriptor file containing information descriptive of an instruction set of said target microprocessor, wherein information about a starting position of at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set, and wherein the descriptor file reflects changes in the instruction set as the changes occur," as recited in claim 5. Furthermore, the cited references do not teach or suggest "automatically tracking the changes in the instruction set by acquiring the data from the descriptor file thereby a hard-coding for the assembling is reduced; constraining the directly transliterated machine language to conform to an architecture of said instruction set, thereby assembling the machine language program for the target microprocessor, using the data acquired from the descriptor file; and transferring the selected data acquired from said descriptor file directly to a linker to enable the linker to perform operations on the constrained machine language using the selected data," as also recited in claim 5.

In view of the foregoing, claim 5 patentably distinguishes over Hadjiyiannis, Vos, and Edmonds, either alone or in combination.

Claim 6 depends from claim 5 and is allowable for at least the same reasons.

Accordingly, withdrawal of the rejection of claims 5 and 6 is respectfully requested.

C. Independent Claim 7

Claim 7, as amended, recites:

A method of preparing a program executable on a target microprocessor comprising:
capturing data from an instruction set of said target microprocessor thereby forming a descriptor file containing information descriptive of said instruction set, wherein information about a starting position of at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction

set is derived and stored as constraints due to the instruction set, and wherein the descriptor file reflects changes in the instruction set as the changes occur;

providing assembly language instructions for said target microprocessor;

translating each assembly language instruction into a corresponding machine language output;

deriving information from the assembly language instructions and acquiring data from the descriptor file using the information, wherein the data is representative of the constraints due to the instruction set;

automatically tracking the changes in the instruction set by acquiring the data from the descriptor file thereby a hard-coding to track the changes is reduced;

using the data acquired from said descriptor file, constraining the machine language output to conform to an architecture of said instruction set; and

transferring the selected data acquired from said descriptor file directly to a linker to enable the linker to perform operations on the constrained machine language using the selected data.

(Emphasis added).

As should be clear from the above discussion, Hadjiyiannis, Vos, and Edmonds do not teach or suggest all of the limitations of claim 7. Specifically, none of the cited references teaches or suggests "capturing data from an instruction set of said target microprocessor thereby forming a descriptor file containing information descriptive of said instruction set, wherein information about a starting position of at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set, and wherein the descriptor file reflects changes in the instruction set as the changes occur," as recited in claim 7. Furthermore, the cited references do not teach or suggest "automatically tracking the changes in the instruction set by acquiring the data from the descriptor file thereby a hard-coding to track the changes is reduced; using the data acquired from said descriptor file, constraining the machine language output to conform to an architecture of said instruction set; and transferring the selected data acquired from said descriptor file directly to a linker to enable the linker to perform operations on the constrained machine language using the selected data," as also recited in claim 7.

In view of the foregoing, claim 7 patentably distinguishes over Hadjiyiannis, Vos, and Edmonds, either alone or in combination.

Accordingly, withdrawal of the rejection of claim 7 is respectfully requested.

D. Independent Claim 8

Claim 8, as amended, recites:

> A method of preparing a program executable on a target microprocessor, comprising:
>
> providing plural program modules, at least one of said modules having one or more instructions including external symbols, wherein external symbols have values which cannot be determined without reference to another program module;
>
> providing a descriptor file containing information descriptive of an instruction set of said target microprocessor, wherein information about a starting position of at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set, and wherein the descriptor file reflects changes in the instruction set as the changes occur;
>
> translating assembly language instructions into machine language wherein the translation comprises:
>
> directly transliterating the assembly language instructions into machine language;
>
> deriving information from the assembly language instructions and acquiring data from said descriptor file using the information, wherein the data is representative of the constraints due to the instruction set and includes selected data comprising at least one encoding function for the at least one instruction due to changes in the instruction set;
>
> *automatically tracking the changes in the instruction set by acquiring the data from the descriptor file thereby a hard-coding to track the changes is reduced;* and
>
> constraining the directly transliterated machine language to conform to an architecture of said instruction set using the data acquired from the descriptor file;
>
> transferring the selected data acquired from said descriptor file directly to a linker to enable the linker to perform operations on the constrained machine language using the selected data; and
>
> binding external symbols to addresses using the at least one encoding function from the selected data acquired from said descriptor file, thereby preparing the program executable on the microprocessor.
>
> (Emphasis added).

As should be clear from the above discussion, Hadjiyiannis, Vos, and Edmonds do not teach or suggest all of the limitations of claim 8. Specifically, none of the cited references teaches or suggests "providing plural program modules, at least one of said modules having one or more instructions including external symbols, wherein external symbols have values which cannot be

determined without reference to another program module; providing a descriptor file containing information descriptive of an instruction set of said target microprocessor, wherein information about a starting position of at least one bit field and a number of bits available for the at least one bit field for at least one instruction in the instruction set is derived and stored as constraints due to the instruction set, and wherein the descriptor file reflects changes in the instruction set as the changes occur," as recited in claim 8. Furthermore, the cited references do not teach or suggest "automatically tracking the changes in the instruction set by acquiring the data from the descriptor file thereby a hard-coding to track the changes is reduced; and constraining the directly transliterated machine language to conform to an architecture of said instruction set using the data acquired from the descriptor file; transferring the selected data acquired from said descriptor file directly to a linker to enable the linker to perform operations on the constrained machine language using the selected data; and binding external symbols to addresses using the at least one encoding function from the selected data acquired from said descriptor file, thereby preparing the program executable on the microprocessor," as also recited in claim 8.

In view of the foregoing, claim 8 patentably distinguishes over Hadjiyiannis, Vos, and Edmonds, either alone or in combination.

Accordingly, withdrawal of the rejection of claim 8 is respectfully requested.

CONCLUSION

In view of the foregoing amendments and remarks, this application should now be in condition for allowance. A notice to this effect is respectfully requested. If the Examiner believes that the application is not in condition for allowance, the Examiner is requested to call the Applicant's representative at the telephone number indicated below to discuss any outstanding issues relating to the allowability of the application.

If this response is not considered timely filed and if a request for an extension of time is otherwise absent, Applicants hereby request any necessary extension of time. If there is a fee occasioned by this response, including an extension fee, please charge any deficiency to Deposit Account No. 23/2825, Reference No. S1022.80572US00.

Dated: November 17, 2008

Respectfully submitted,

By _____
James H. Morris
Registration No.: 34,681
WOLF, GREENFIELD & SACKS, P.C.
Federal Reserve Plaza
600 Atlantic Avenue
Boston, Massachusetts 02210-2206
617.646.8000

x11/16/2008x